FM TOWNS

FUJITSU FM SERIES PERSONAL COMPUTER

アプリケーション開発キット
ユーザーズガイド



FUJITSU

FUJITSU

81SP-0071-1

FM TOWNS
アプリケーション開発キット
ユーザーズガイド

富士通株式会社

ごあいさつ

このたびは「FM TOWNS 386 | ASM TOOL KIT V1.1」をお買い上げいただき、まことにありがとうございます。

本書には、開発キットで提供されるソフトウェアやマニュアルの概要、FM TOWNSでアプリケーションを開発する環境や手順についての概要など、本開発キットの全体を見渡せる情報が記載されています。

本マニュアルには、"外国為替及び外国貿易管理法"に基づく特定技術が含まれています。したがって、本マニュアルまたはその一部を輸出する場合には、同法に基づく許可が必要とされます。

1989年10月

■開発キットのマニュアル構成と本書の位置づけ

FMTOWNS 上で、C言語やアセンブラでのソフトウェア開発を可能にするため、富士通では「FMTOWNS 386 | ASM TOOL KIT V1.1」と「FMTOWNS High C™ Compiler V1.4」の2つの関連した開発キットを用意しております。

以下に示すのはこれら2つの開発キットに添付されているマニュアルです。各マニュアルの関連性を考え、本書をお読みください。

製品名称：　FMTOWNS 386 | ASM TOOL KIT V1.1

マニュアル	内容
アプリケーション開発キット ユーザーズガイド	開発キットで提供されるソフトウェアやマニュアルの概要、FMTOWNS でアプリケーションを開発する環境や手順についての概要など全体を見渡せる情報が記載されています。
386 | DOS-Extender™ ユーザーズマニュアル	386 | DOS-Extender™のもとで 80386ネイティブモードのプログラムを実行する方法について説明し、さらに 80386ネイティブモード環境についても詳細に解説しています。
386 | ASM TOOL KIT ユーザーズマニュアル	FMTOWNS 上でアセンブラを使用してアプリケーションプログラムを開発するユーザのために、開発の過程で必要な各ソフトウェアの使用方法を詳細に解説しています。 次の4つの部で構成されています。 第1部　アセンブラ　386 | ASM 第2部　リンカ　386 | LINK 第3部　シンボリックデバッガ　386 | DEBUG 第4部　ライブラリアン　386 | LIB

注）網掛けで示されたマニュアルが本書です。

TownsBIOS
リファレンス

FMTOWNS の持つBIOS機能について、リファレンス形式で詳細に解説しています。
BIOSの機能により、次の6つの章に分類されています。

1 グラフィックBIOS　　4 フォントBIOS
2 マウスBIOS　　5 CD-ROM BIOS
3 スプライトBIOS　　6 サウンドBIOS

製品名称： FMTOWNS High C™ Compiler V1.4

High C™ Compiler
ユーザーズマニュアル

FMTOWNS でHigh C™ Compiler を使用してアプリケーションプログラムを開発するユーザのために、プログラム開発過程、High Cの文法、High Cの標準ライブラリについて解説していまます。
次の3部で構成されています。

第1部　プログラマーズガイド
第2部　ランゲージリファレンス
第3部　標準ライブラリ

Cライブラリ
リファレンス

FMTOWNS 専用のCライブラリについて、その機能や、使用方法、使用例などをリファレンス形式で詳細に解説しています。
ライブラリ関数の機能により、次の8つの章に分類されています。

1 グラフィック　　5 CD-ROM
2 マウス　　6 サウンド
3 スプライト　　7 EUPHONY ドライバ
4 フォント　　8 キーボード

目次

アプリケーション開発キット
ユーザーズガイド

図表一覧

第1章 概要

1.1 開発キットの種類

FMTOWNS上で、アセンブラやC言語でのソフトウェア開発を可能にするため、富士通では次の2種類のパッケージを用意しております。これらは、FMTOWNSのアプリケーションプログラムを効率よく開発するためのソフトウェアツールを統合した、プログラマのためのキットです。

「FMTOWNS　386|ASM TOOL KIT」パッケージ

「FMTOWNS 386|ASM TOOL KIT」パッケージは、FMTOWNSでアセンブラを使用して、アプリケーションプログラムを開発しようとするユーザのための開発パッケージです。パッケージにはアセンブラ、リンカ、シンボリックデバッガ、ライブラリアンのソフトウェアとマニュアルが含まれています。
（パッケージ中のアセンブラ386|ASMは、80386マイクロプロセッサのためのアセンブラです。）

「FMTOWNS　High C ™ Compiler」パッケージ

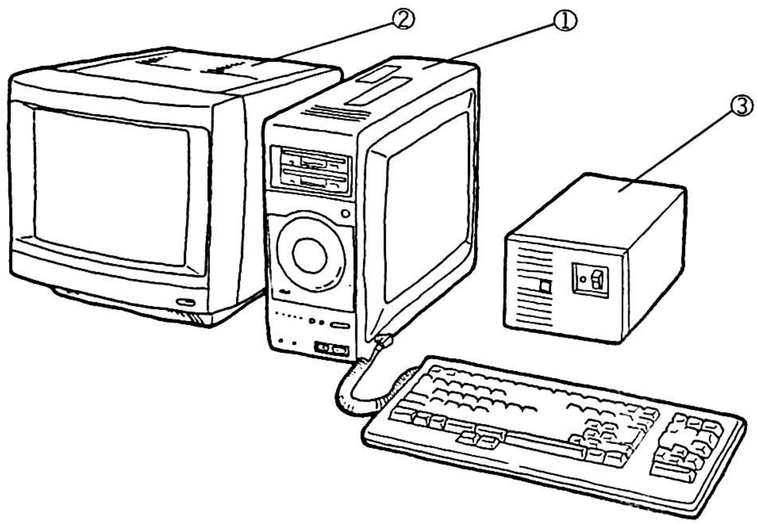
「FMTOWNS High C™ Compiler」パッケージは、FMTOWNSでC言語を使用して、アプリケーションプログラムを開発しようとするユーザのためのパッケージです。パッケージにはコンパイラと標準ライブラリのソフトウェアとマニュアルが含まれています。

このパッケージには、プログラム開発に必要なリンカやデバッガなどのツールが含まれていませんので、「FMTOWNS 386|ASM TOOL KIT」パッケージもあわせてご使用ください。

1.2 開発環境

FMTOWNSで開発キットを使用してアプリケーションプログラムを開発するには、次のハードウェアとソフトウェアが必要です。

ハードウェア環境



① FMTOWNS ……………………………… 本体
- RAM ……………………………… 2 MB
- フロッピィディスクドライブ（FPD）…… 2台
- キーボード ……………………………… 1台

② ディスプレイ ……………………………… 1台

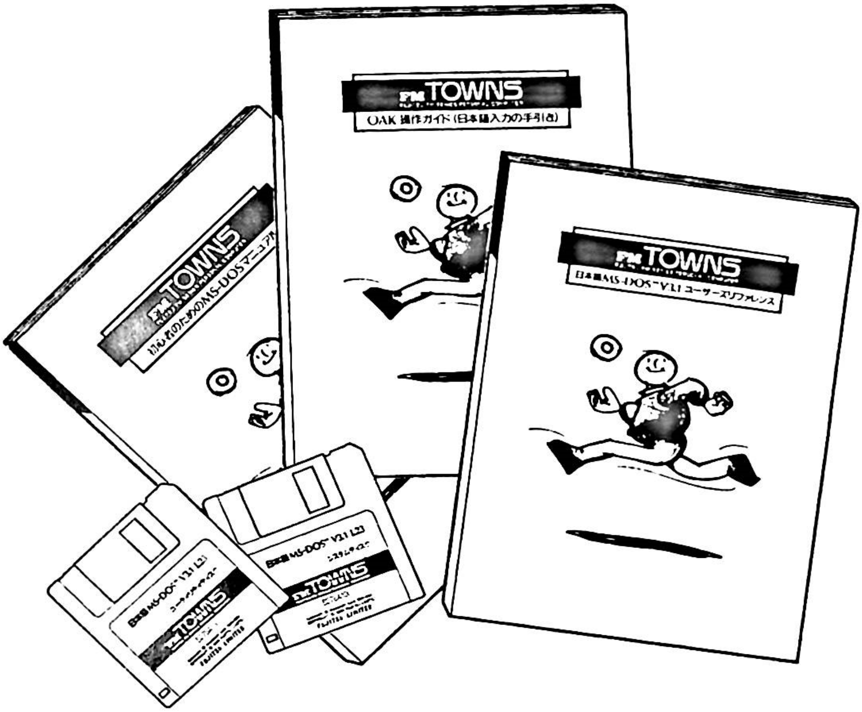
③ ハードディスク（20MB以上）…………… 1台

（開発効率を向上させ快適な開発環境にするには、ハードディスクが不可欠です。）

図1.1　ハードウェア環境

ソフトウェア環境

開発キットには含まれていませんが、アプリケーションプログラムを開発する環境として、次のソフトウェアが必要になります。



① 日本語ＭＳ－ＤＯＳ™Ｖ３．１
FMTOWNSで、市販されている多くのMS-DOSで動作するプログラムを実行するためにも「日本語MS-DOS™V3.1」が必要です。

② エディタ（ＥＤＬＩＮなど）
ソースプログラムのファイルを作成するためにエディタが必要になります。
（ＥＤＬＩＮは日本語MS-DOS™V3.1に含まれています。）

図1.2　ソフトウェア環境

1.3 ソフトウェア構成

「386|ASM TOOL KIT」パッケージ

このパッケージには、次の図に示すように、386|ASM 、386|LINK、386|DEBUG、386|LIB の4つのプログラムが含まれています。プログラムは 3.5インチディスク1枚で提供されます。

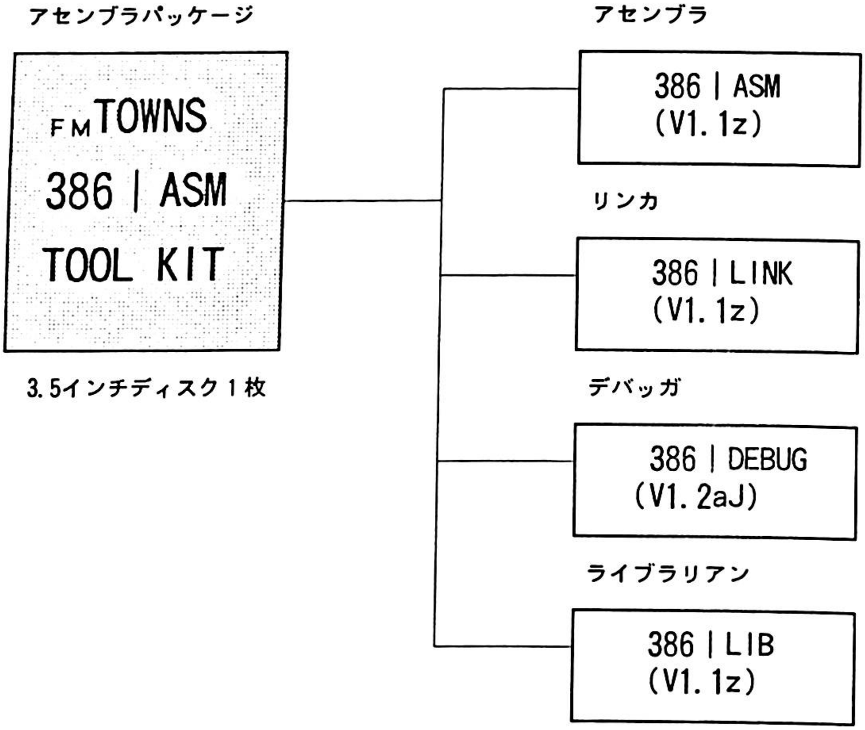


図1.3 「386|ASM TOOL KIT」ソフトウェア構成

386|ASM (V1.1z)

386|ASM は、80386 マイクロプロセッサ用のアセンブラです。1つまたは複数のアセンブリ言語ソースコードファイルを、1つのオブジェクトモジュールにアセンブルするために使用します。80386 用の命令コードをフルサポートし、80387 コプロセッサ命令もサポートしています。

386 | ASM の命令ニーモニックは、インテル／ＭＡＳＭと互換性があり、アセンブラ命令およびマクロ命令はMASMと互換性があります。さらに、オブジェクトモジュールもインテル／マイクロソフトと互換性があります。

386 | LINK (V1.1z)

386 | LINKは、386 | ASM やHigh C™で出力したオブジェクトモジュールのリンカです。アセンブラまたはコンパイラで作成された別々のオブジェクトモジュールを、1つの 80386アプリケーションに組み合わせるのに使用します。386 | LINKで出力される実行モジュールの形式は、次のようになっています。

・MS-DOSの場合は、ファイルの拡張子が".EXE"で出力されます。
・386ネイティブモードの場合は、ファイルの拡張子が".EXP"で出力されます。
・インテルアブソリュートHEX形式、これはROM化などに使用します。
　また、SYMBOL/MAPなどを出力することができます。

386 | DEBUG (V1.2aJ)

386 | DEBUGは、80386マイクロプロセッサ用のシンボリックデバッガです。386ネイティブモードまたはリアルモードのどちらかで実行するように作成されたプログラムをデバッグするために使用します。386ネイティブモードのプログラムは、386 | DOS-Extender™の環境でデバッグし、リアルモードのプログラムは、標準MS-DOSの環境でデバッグします。

386 | DEBUGは、386 ネイティブモードとリアルモードの両モードにおいて 80386の全レジスタの表示と変更が可能であり、さらに 80386の全命令を逆アセンブルすることができます。また、メモリの表示、変更、移動、比較を行うコマンド、プログラム中にブレークポイント、ウォッチポイントをセットするコマンドおよびシングルステップをセットするコマンドなどがあり、機能性、操作性ともに優れています。

また、386 | DEBUGのコマンド名称と一般形式は、MS-DOSで提供されるSYMDEBユーティリティにあるコマンド名称および一般形式と同様になっています。

386 | LIB (V1.1z)

386 | LIB は、386 | LINKで生成したオブジェクトモジュールを、ライブラリファイルに登録したり、置き換え、追加、削除、バックアップなどを行うときに使用します。また、ライブラリのマップを出力することもできます。

「High C™ Compiler」パッケージ

このパッケージには、次の図に示すように、High C™コンパイラ(V1.4)、High C™ライブラリ、FMTOWNS用Cライブラリなどのソフトウェアが含まれています。プログラムは 3.5インチディスク2枚で提供されます。

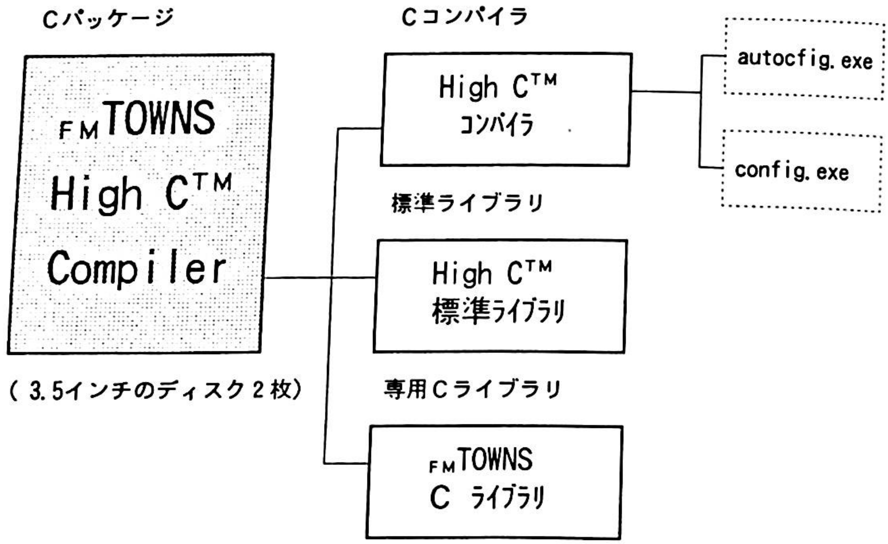


図1.4 「High C™ Compiler」ソフトウェア構成

High C™ Compiler

High C™は、80386 マイクロプロセッサ上のMS-DOSのもとで動作するC言語コンパイラで、言語仕様は"ANSI X3J11"の規格に準拠しています。生成されたプログラムは、80386 マイクロプロセッサ用のネイティブコードで、386 | DOS-Extender™のもとで動作します。

メモリモデルはすべてスモールモデルで実現しており、32ビットオフセットのアドレッシングや、32ビットサイズのint 型の定義などが可能です。

拡張機能としては、pragma機能、符号無し定数の指定、switch/case 文の範囲指定など独自のものをもっています。さらに、レジスタ割付けの最適化、共通式の削除／分岐命令の最適化などオブジェクトの効率化がはかられています。

ユーティリティソフトウェアとしては、ランタイムライブラリ、ヘッダファイル、クロスリファレンス機能が含まれています。

また、コンパイラを構成するためにautocfig.exeとconfig.exeプログラムを提供します。

autocfig.exe

インストールプログラム（INSTALL.EXE）の中には、autocfigプログラム（automatic configuration program）が組み込まれており、インストールプログラムを実行することにより自動的に、現在使用可能なメモリに対応したツリーページ（-tpages）を設定し、現在使用可能メモリ値と-tpages値を表示します。メモリ構成を変更する場合は、コンパイラ（hc386.exe）を含むディレクトリでautocfigプログラムを実行してください。

メモリ不足（out of memory)が発生した場合は、ツリーページの数を約20減らさなければなりません。また、コンパイラに対してメモリ制限を通知することができ、これにより、コンパイラは少し実行速度が落ちますがメモリ保護を行います。たとえば、

```
hc386 program.c -tpages 20 -cram
```

と指定すると、20ツリーページを持ちメモリ制限を想定するようにコンパイラを構成することができます。また、 -ansiオプションを指定すると、さらに15KBのメモリを必要とします。

config.exe

各種のコンパイラ・デフォルトは hc386 program.c -off Warnのように指示されたコンパイラの呼び出しライン上或いは、pragma off(Warn); のようなプログラムのソース・コード中であろうとディレクティブを介して変更ができます。

configプログラムは、コンパイラのデフォルトを変更でき、プログラムがコンパイルされてもデフォルトは、configで再度変更されるまで保持されます。また、実行後は自己ドキュメントを行い、各オプションのデフォルトを表示します。
各オプションについては、「High C™ Compiler ユーザーズマニュアル」の「第1部　プログラマーズガイド」を参照してください。

このプログラムで、Cコンパイラにより決められたオプションのデフォルト値を変更するには、文字(A～O)のオプションを入力します。オプションについてさらに知りたい方は、その後に"?"を入力してください。

```
N?
```

終了する場合は、リターンキーを押します。続いて再度確認があるので、設定する場合はY またはy を入力してください。デフォルトのままで終了する場合は、リターンキーを押してください。

ユーティリティソフトウェア

コンパイルおよびランタイム・ランプラリには、UNIX™のように有効なユーティリティプログラムが提供しています。ユーティリティの中には、ディレクトリの再帰的リスト（ls）やストリングのサーチを行うプログラム（fgrep）などが含まれています。fgrep は非常に高速なため、大量印刷用のペーパー・クロス・リファレンスなどでは効果的な代替手段となります。

各ユーティリティについては、オンライン・ドキュメントを参照してください。

コンパイラ（HC386.EXE)を実行する時にパラメータを省略した場合は、標準入力からプログラムの入力ができます。なお、プログラムの入力の終わり(end_of_program)は、control-zを入力します。

High C™ライブラリ

High C™ライブラリは、ANSI規格のC標準ライブラリに準拠したライブラリをヘッダファイルとして提供しています。詳細については、High C™ Compiler ユーザーズマニュアル「第3部　標準ライブラリ」を参照してください。また、ランタイムライブラリも提供しています。

ランタイムライブラリ

ランタイムライブラリは、いくつかのオブジェクトファイルとライブラリファイルから成っています。オブジェクトファイルは、High C™の中のsmall ディレクトリ中にあります。以下は、ファイルのリストです。

hcc.lib	80387 コプロセッサ(C) を想定したランタイムライブラリ (hc)
hce.lib	80387 を想定してはいない(emulator の e) が、あれば使います
stkdmp.obj	特定の事後分析トレースを受け取るのに用います
heapl.obj	〃
c_heap.obj	必要に応じてheapマネージャを削除するのに用います
c_close.obj	メモリ要件を軽減します
c_scanf.obj	〃
c_printf.obj	〃

詳細については、High C™ Compiler ユーザーズマニュアル「第2部　プログラマーズガイド」の「コンパイル済プログラムのリンク」の章を参照してください。

FMTOWNS Cライブラリ

FMTOWNS　Cライブラリは、FMTOWNS専用のライブラリで、次のものが提供されています。

・グラフィックライブラリ
・マウスライブラリ
・スプライトライブラリ
・フォントライブラリ
・ＣＤ－ＲＯＭライブラリ
・サウンドＢＩＯＳライブラリ
・ＥＵＰＨＯＮＹドライバ
・キーボードインターフェイスライブラリ

1.4 マニュアル構成

「386｜ASM TOOL KIT」パッケージ

このパッケージには、以下に示す4冊のマニュアルが含まれています。

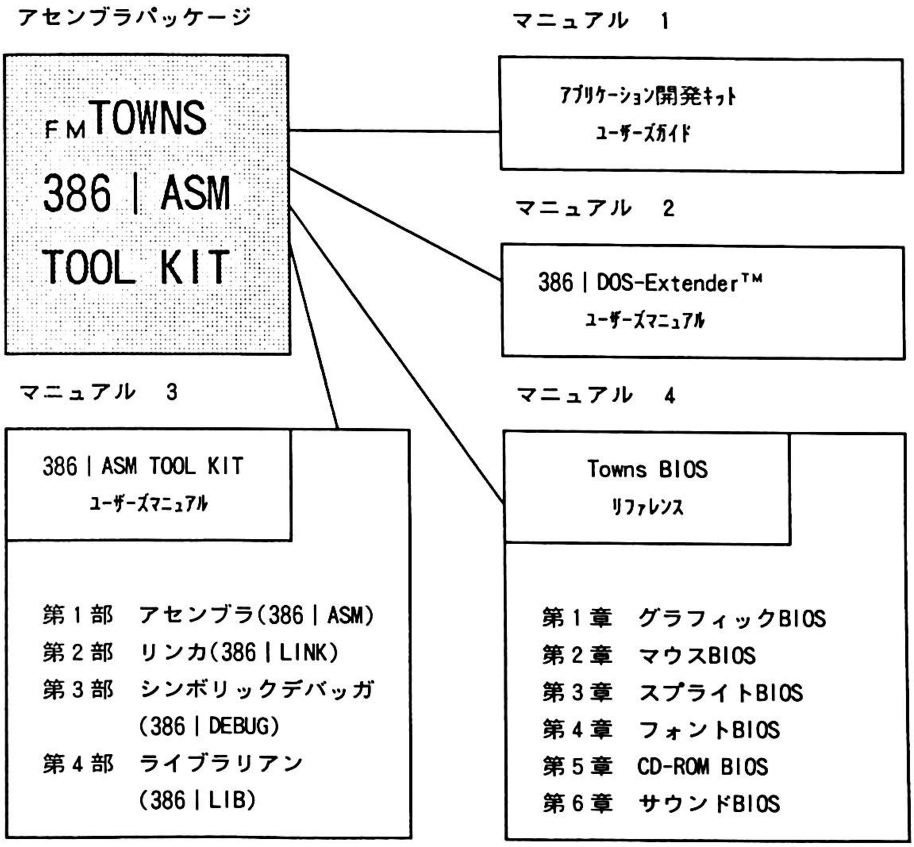


図1.5 「386｜ASM TOOL KIT」マニュアル構成

アプリケーション開発キットユーザーズガイド

開発キットで提供されるソフトウェアやマニュアルの概要、FM TOWNS でアプリケーションを開発する環境や手順についての概要など全体を見渡せる情報が記載されています。

386 | DOS-Extender™ユーザーズマニュアル

386 | DOS-Extender™ のもとで 80386ネイティブモードのプログラムを実行する方法について説明し、さらに 80386ネイティブモード環境についても詳細に解説しています。

386 | ASM TOOL KITユーザーズマニュアル

FMTOWNS 上でアセンブラを使用してアプリケーションプログラムを開発するユーザのために、開発の過程で必要な各ソフトウェアの使用方法を詳細に解説しています。次の4つの部で構成されています。

第1部　アセンブラ　(386 | ASM)

386 | ASM はインテル 80386マイクロプロセッサ用アセンブラです。第1部では386 | ASM でアセンブラソースコードをアセンブルする方法を解説します。

第2部　リンカ　(386 | LINK)

386 | LINKはインテル 80386マイクロプロセッサ用のリンカです。第2部では386 | LINKを使ってアセンブラまたはコンパイラで作成した別々のオブジェクトモジュールを、単一の 80386アプリケーションに組み合わせる方法を解説します。さらに各種コマンドスイッチをリファレンス形式で解説します。

第3部　シンボリックデバッガ　(386 | DEBUG)

386 | DEBUG はインテル 80386マイクロプロセッサ用のシンボリックデバッガです。第3部では、386 | DEBUG を使ってプログラムをデバッグする方法を解説します。さらにデバッガコマンドをリファレンス形式で解説します。

第4部　ライブラリアン　(386 | LIB)

386 | LIBは、386 | LINKで生成されたオブジェクトファイルからライブラリファイルを作成するためのライブラリアンプログラムです。第4部では、386 | LIBの使い方と各種コマンドラインスイッチを解説します。

Towns BIOSリファレンス

FMTOWNS の持つBIOS機能について、リファレンス形式で詳細に解説しています。BIOSの機能により、次の6つの章に分類されています。

1 グラフィックBIOS	4 フォントBIOS
2 マウスBIOS	5 CD-ROM BIOS
3 スプライトBIOS	6 サウンドBIOS

「High C™ Compiler 」パッケージ

このパッケージには、下図に示す２冊のマニュアルが含まれています。

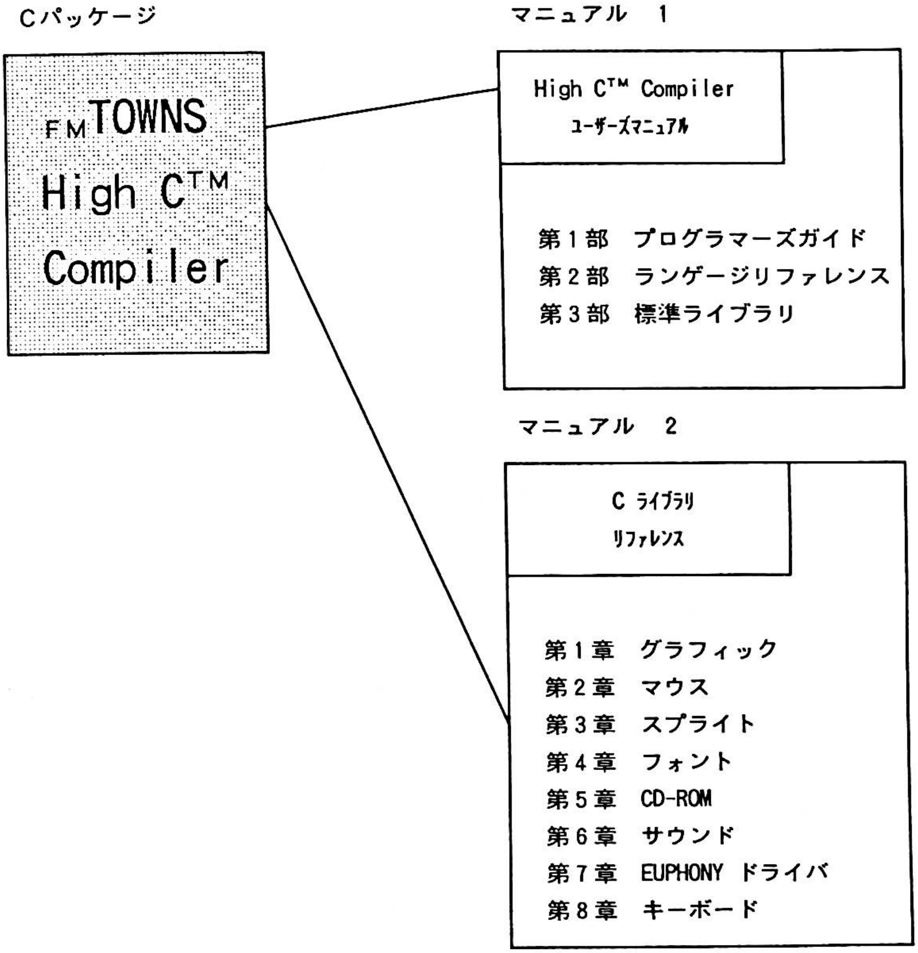


図1.6 「High C™ Compiler」マニュアル構成

High C™ Compiler ユーザーズマニュアル

FMTOWNS でHigh C™ Compiler を使用してアプリケーションプログラムを開発するユーザのために、プログラム開発過程、High C™の文法、標準ライブラリについて解説しています。次の３部で構成されています。

第１部　プログラマーズガイド
High C™ Compiler で効果的にプログラムを開発するために必要な事項を解説しています。
扱っている内容は、コンパイル・リンク・実行の方法、コンパイラ制御およびオプションの使い方、記憶域マッピング、デバッギング、実行のしくみ、省略値および制限条件、メモリモデルなどです。

第２部　ランゲージリファレンス
Ｃ言語の文法について解説しています。初心者のためのＣ言語の入門書ではありません。Ｃ言語の、ハードウェアに依存しない事項に関する疑問に答えることを目的に書かれています。

第３部　標準ライブラリ
High C™標準ライブラリのリファレンスマニュアルです。構成、形式、内容などの解説に続き、ソースプログラム中のヘッダファイル、ライブラリで用いるパラメータの受渡し規則、ライブラリで使用するネーミング規則などについて解説しています。
また、stdio.h を初めとする各ヘッダファイルについて、その中に含まれているマクロの機能と使用方法を記述しています。

Ｃライブラリリファレンス

FMTOWNS 専用のＣライブラリについて、その機能や、使用方法、使用例などをリファレンス形式で詳細に解説しています。
ライブラリ関数の機能により、次の８つの章に分類されています。

1 グラフィック	5 CD-ROM
2 マウス	6 サウンド
3 スプライト	7 EUPHONY ドライバ
4 フォント	8 キーボードインターフェイス

第2章 動作環境の構築

これより、ハードディスクの構築を行い、各開発キットのインストールを行います。ハードディスクの区画設定は開発環境、TownsOS（TownsMENU）、日本語MS-DOS™、ＷＯＲＫの４つに区分けします。ここで、開発環境について説明します。

FMTOWNS でのアプリケーション開発システムとして開発環境を提供しています。この開発環境は、日本語MS-DOS™の一部を除くすべてのコマンドを使用することができるネイティブモードシステムで、ネイティブモードで動作可能なアプリケーションを開発することができます。よって、ソースプログラムの作成・編集、アセンブル（またはコンパイル）、リンク、デバッグ、実行までのすべての処理を行うことができます。

ただし、開発環境の使用においては、次の制限事項があります。

ＫＥＹＳＥＴが使用できない　(MS-DOSコマンド) テンプレート機能が使用できない 日本語入力（かな漢字変換）機能が使用できない

ハードディスクの区画が終了したら、それぞれの区画に次の各システムをインストールします。

①TownsOS（TwonsMENU）	TownsOSシステム
②開発環境	386｜ASM TOOL KIT、High C Compiler
③日本語MS-DOS™	日本語MS-DOS™システム
④ＷＯＲＫ	ソースプログラム等の保存用（ワーク領域）

【注　　意】 開発環境でMS-DOSコマンドを使用する場合は、PATH設定を行うか、各コマンドをコピーする必要があります。

2.1 各システムの切り換え

インストールを行う前に各システムの切り換えロジックを説明します。

FMTOWNSで、開発環境、TownsOSシステム、日本語MS-DOS™システムへ移行をしたい場合は、REIPL コマンドによるシステムの再起動が必要となります。

日本語MS-DOS™の "REIPL" コマンドでは、起動するドライブが指定できます。

以下にFMTOWNSがシステムを探す順序と、各システムの切り換えロジックを示します。

●FMTOWNSがシステムを探す順序

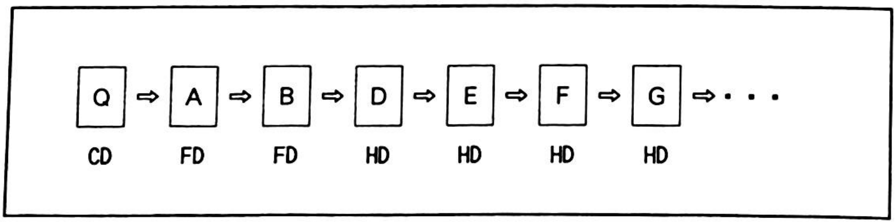


図2.1　FMTOWNSがシステムを探す順序
（起動ドライブをQ：に設定した場合）

●システムの切り換えロジック

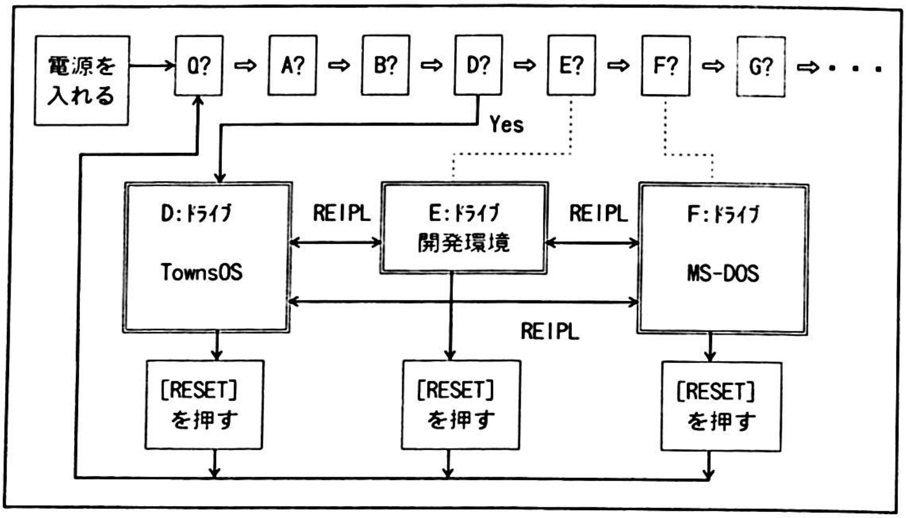


図2.2　システムの切り換えロジック

2.2 日本語MS-DOS™のインストール

開発キットの"386|ASM"および"High C™"を使用する前に、ハードディスクの区画を設定して、開発環境およびTownsOS、MS-DOSの起動に必要なファイルをハードディスクへコピーします。以下にハードディスクの区画を設定する手順について説明します。既に、ハードディスクの区画を設定している場合は、必要な区画のみを設定して、インストールを行ってください。

ハードディスクの区画について

ハードディスクユニットの区画は、次の表のように設定します。

表2.1 ハードディスクの区画（40MBハードディスクの場合）

区画	区画名	ＯＳ種別	起動／非起動	容量	割合
1*	ＴｏｗｎｓＯＳ	ＭＳ－ＤＯＳ	非起動	１０．０	
2*	開発環境	ＭＳ－ＤＯＳ	非起動	１０．０	
3*	ＭＳ－ＤＯＳ	ＭＳ－ＤＯＳ	起動	１０．０	
4	ＷＯＲＫ	ＭＳ－ＤＯＳ	非起動	１０．０	

* の区画は、必ず指定してください。

【注　意】 開発環境の容量は、最低３MB以上に設定してください。
MS-DOSとTownsOSの容量は、最低２MB以上に設定してください。

【制約事項】 DISKPAT について：
日本語MS-DOS™V3.1の１区画の容量は、127.1MByte以上設定することはできません。

ハードディスクの区画設定と日本語MS-DOS™のインストール

以下に示すのはハードディスクに区画を設定し、日本語MS-DOS™をインストールする手順です。

[STEP 1] ハードディスクをFMTOWNSに接続します。接続方法は、各ハードディスクユニットに添付のマニュアルを参照してください。

[STEP 2] 日本語MS-DOS™V3.1システムディスクを“ドライブＡ”にセットして、起動します。

[STEP 3] 日本語MS-DOS™の“DISKPAT ”コマンドを使って、ハードディスクの区画を表2.1のように設定します。このとき、区画番号３のMS-DOSを必ず“起動”に設定してください。設定の終了時に、区画３にのみMS-DOSシステムを“複写”してください。

[STEP 4] 日本語MS-DOS™の“SETUP”コマンドを使って、ドライブ名を設定します。ドライブ名は、区画番号１から順にＤ：、Ｅ：、Ｆ：、Ｇ：となるように設定してください。“ドライブＣ：”は、TownsOSシステムの予約ドライブに指定されているため、設定しないでください。

[STEP 5] 日本語MS-DOS™V3.1システムディスクを“ドライブＡ”から再起動してください。[STEP 4]で設定したドライブ番号が確定されます。

[STEP 6] 日本語MS-DOS™の“FORMAT”コマンドを使って、ハードディスクの“ドライブＧ：（ＷＯＲＫ）”を初期化します。設定していない場合は、次へ進んでください。“ドライブＤ：”、“Ｅ：”は、TownsOS システムをコピーするため、そのままにしておいてください。

[STEP 7] 日本語MS-DOS™の“COPY”コマンドを使って、MS-DOSシステムフロッピィディスクの中のファイルをすべて“ドライブＦ：”にコピーします。

これで日本語MS-DOS™のインストールを終了しました。
次に、開発環境およびTownsOSのインストールを行ってください。

2.3 開発環境のインストール

以下に示すのは開発環境をインストールするまでの手順です。次の操作方法は、Towns MENU V1.1 L20を前提としています。

[STEP 1]　CD-ROMドライブにFMTOWNSシステムソフトウェアをセットしてTownsMENUを起動してください。

[STEP 2]　TownsMENUの「ディスク」から「初期化」を選択して、“TownsOSを複写する”をクリックして、“ドライブＥ：”を初期化します。初期化後に開発環境の起動に必要なファイルがコピーされます。

[STEP 3]　TownsMENUの「ファイル」から「表示形式」の「ＡＢＣ順表示」を選択し、FMTOWNSシステムソフトウェアの中の全ファイルを表示します。

[STEP 4]　“autoexec.bat”ファイルをクリックして選択し、TownsMENUの「ファイル」から「削除」を選択して、ファイルを削除してください。
“autoexec.bat”ファイルは、「386 | ASM TOOL KIT」のインストール時にコピーされます。

開発環境のセットアップは、後の「2.5 開発環境の起動」へ続きます。

2.4 TownsOSのインストール

以下に示すのはTownsOSをインストールし、ハードディスクを起動するまでの手順です。次の操作方法は、TownsMENU V1.1 L20を前提としています。

[STEP 1]　TownsMENUの「ディスク」から「初期化」を選択して、“TownsOSを複写する”をクリックして、“ドライブD：”を初期化します。初期化後にTownsMENUの起動に必要なファイルがコピーされます。

これで、TownsOSのインストールは終了しました。

[STEP 2]　TownsMENUから「設定」の「起動ドライブ」を選択して、上記で設定したハードディスクユニットを設定します。

[STEP 3]　TownsMENU右上の“EXIT”アイコンをクリックした後、「リセット」を選択してハードディスクから日本語MS-DOS™を起動します。

[STEP 4]　REIPL.COMファイルを“ドライブD：”、“E：”にコピーします。

【参　考】　TownsMENUにおいてREIPLコマンドをアイテム登録するには、次の操作を行ってください。

[STEP 1]　TownsMENUを起動します。

[STEP 2]　TownsMENUの「ファイル」から「表示形式」の「ＡＢＣ順表示」を選択し、FMTOWNSシステムソフトウェアの中の全ファイルを表示します。

[STEP 3]　“REIPL.COM”ファイルをクリックして選択します。

[STEP 4]　「ファイル」から「アイテム登録」を選択します。

[STEP 5]
- “登録／変更”ボタンを選択
- “パラメータ”指定において起動したいドライブを(例:“D”)入力します。
- “アイコン”と“タイトル”を指定します。

[STEP 6]　左上のアイコンをクリックしてアイコン表示にして、アイコンが登録されたか確認してください。

2.5 開発環境の起動

開発環境を起動させるには、次の2.6 「386|ASM TOOL KIT」のインストールプログラム（INSTALL.EXE）を実行する必要があります。このインストールプログラムを実行することにより開発環境の起動に必要なファイルをコピーします。

インストールが終了後に、“DISKPAT”コマンドを使って、ハードディスクの区画２の開発環境を“起動”に設定してください。これで、すべての設定は終了しました。本体をリセット（またはREIPL コマンドを使用。例、REIPL D:）することにより開発環境が起動します。

2.6 「386|ASM TOOL KIT」のインストール

インストールディスクに含まれている“INSTALL.EXE ”を実行して、指定のドライブへファイルのインストールを行います。インストールは、インストールディスクを“ドライブＡ：”にセットして、実行してください。

```
A>INSTALL
```

2.7 「High C™Compiler」のインストール

インストール

インストールディスクに含まれている“INSTALL.BAT ”を実行して、指定のドライブへファイルのインストールを行います。インストールは、インストールディスク＃１を“ドライブA：”にディスク＃２を“ドライブB：”にセットして、実行してください。

```
A>INSTALL
```

環境変数の設定

High C™ Compilerでは、ヘッダファイルの場所の指定を行う場合に“IPATH ”を設定することができます。たとえば、 D:¥HC386¥INC¥をサーチするには、次のように指定します。

```
set IPATH=D:¥HC386¥INC¥
```

“IPATH ”の設定方法は、日本語MS-DOS™V3.1のPATH指定と同様にAUTOEXEC.BATファイルに設定することにより、日本語MS-DOS™V3.1システムの起動時に常に設定されます。“IPATH ”の詳しい説明は、「High C™ Compilerユーザーズマニュアル」の中の「プログラマーズガイド」を参照してください。

なお、インストールプログラム実行により、このIPATHを設定することができます。

第3章 プログラムの作成

3.1 プログラム作成の概要

FMTOWNSではTownsOSが 80386のネイティブモードをサポートしており、このネイティブモードに対応した「386|ASM」および「High C™」開発キットが提供されています。したがってプログラムの開発には、これらの開発キットを利用します。以下にプログラム開発の手順の概要を述べます。

ソースファイルの作成

まず開発環境（または日本語MS-DOS™）上でテキスト編集（エディタ）を使用してソースファイルを作成・編集します。

コンパイルとリンク

次に開発環境（または日本語MS-DOS™）上でソースファイルをコンパイルし、リンクし、実行（.EXPなど）ファイルを作成します。

デバッグ

作成された実行ファイルのデバッグは、開発環境上で行います。

FMTOWNSのデバッグは次の２通りの方法があります。

①FMTOWNS 単体によるデバッグ

②端末使用によるデバッグ

動作確認

完成したソフトウェアの動作確認は、次の方法で行います。

【開発環境　　】“.EXP”ファイルは、RUN386　<ファイル名>を使用します。
“.EXE”ファイルは、<ファイル名>をダイレクト入力します。

【ＭＳ－ＤＯＳ　】“.EXE”ファイルは、<ファイル名>をダイレクト入力します。

【ＴｏｗｎｓＯＳ】アイテム登録またはファイルをダブルクリックします。

3.2 開発キットの動作概念

FM TOWNSで開発キットを使用するための動作概念図を示します。これは、それぞれのシステムで実行できる処理の一覧を表した図です。

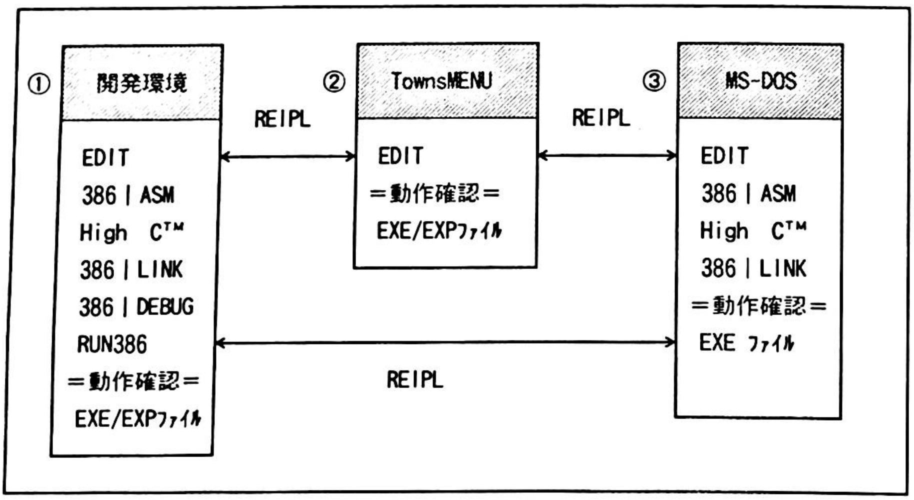


図3.1　開発キットの動作概念

① 開発環境では、次の処理を行うことができます。
- −テキスト編集プログラム（TownsMENU V1.1 L20以降）によるソースプログラムの作成・編集
- −アセンブル（386 | ASM）またはコンパイル（HC386）の実行
- −リンク（386 | LINK）の実行
- −デバッグ（386 | DEBUG）の実行
- −実行（EXE/EXP）ファイルの動作確認（コンソール出力可能）

② ＴｏｗｎｓＭＥＮＵでは、次の処理を行うことができます。
- −テキスト編集プログラム（TownsMENU V1.1 L20以降）によるソースプログラムの作成・編集
- −実行（EXE/EXP）ファイルの動作確認

③ ＭＳ−ＤＯＳでは、次の処理を行うことができます。
- −エディタ(EDLINなど）によるソースプログラムの作成・編集
- −アセンブル（386 | ASM）またはコンパイル（HC386）の実行
- −リンク（386 | LINK）の実行
- −実行（EXE）ファイルの動作確認

3.3 ソース編集、コンパイル、リンク

ソースプログラムの作成から実行プログラムファイルの作成までの作業は、すべて開発環境（または日本語MS-DOS™）上で行います。それぞれのプロセスは、次の図のようになります。

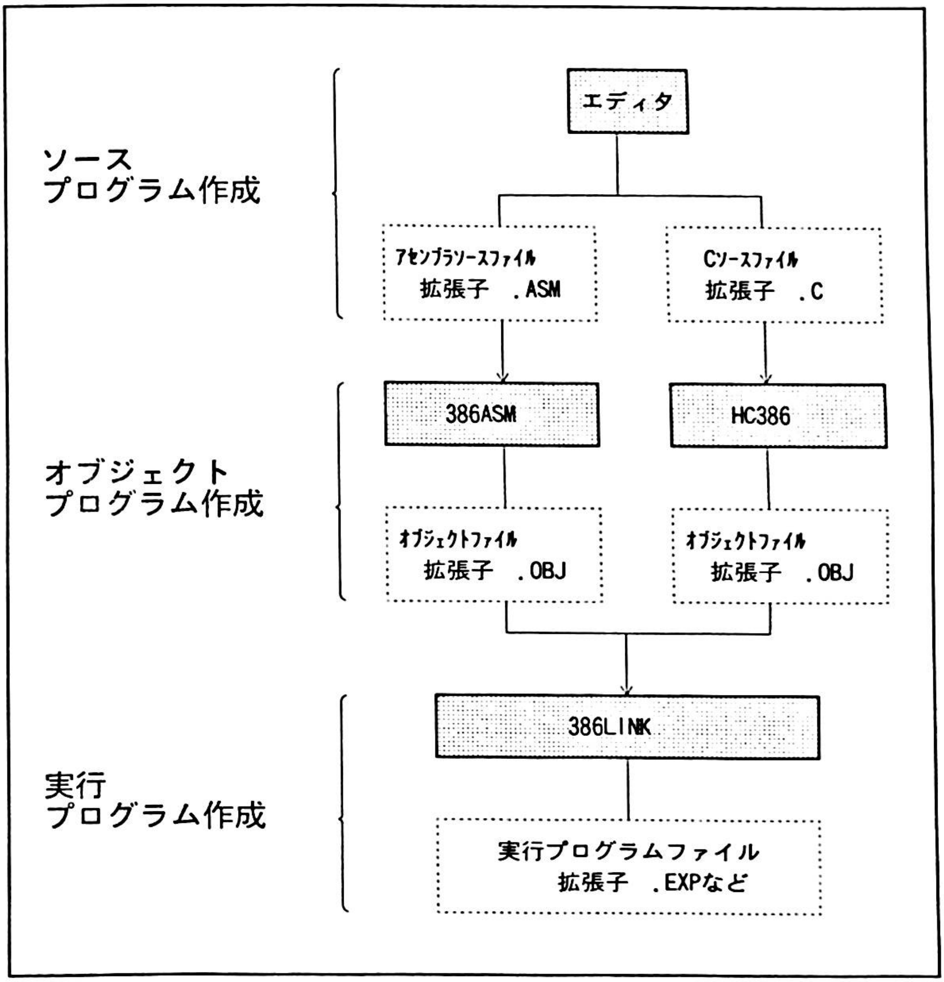


図3.2　ソース編集、コンパイル、リンクのプロセス

3.4 プログラムのデバッグと実行

386|LINKで作成した実行プログラムファイル（拡張子　.EXPなど）は、開発環境上でデバッグし、実行します。不具合がある場合は、エディタを使ってプログラムを修正し、コンパイルまたはアセンブル、リンクしたのち再実行します。

ここでは、開発環境上での386|DEBUGの起動から終了までのプロセスについて説明します。なお、デバッグの方法には次の２通りがあります。

■FMTOWNS 単体によるデバッグ
■端末使用によるデバッグ

■FMTOWNS 単体によるデバッグ

386|DEBUG（デバッガ）の起動は、開発環境を起動した後、次のように入力します。

```
D>386DEBUG  <ファイル名>
```

386|DEBUGが起動されると、画面モードが「640×400ドット・16色・２プレーン」のMS-DOS互換画面になります。

このとき、画面モードを変更するようなプログラムを実行すると、テキスト画面が乱れて文字が読めなくなる場合があります。この場合には、［PF12］キーを押すとMS-DOS互換画面に切り替わり、再び文字が読めるようになります。

デバッグを実行すると、グラフィック画面は表示されますが、テキスト情報が表示されません。そこで、テキスト情報を表示するために画面切り換えプログラム（［PF11］／［PF12］）が用意されています。使用方法は、次のとおりです。

[PF11] と [PF12]

● [PF11] キーによる画面の切り換え

640×400 ドット・16色・2プレーン

・[PF11] キーを押す ⇩

640×480 ドット・16色・2プレーン

・[PF11] キーを押す ⇩

320×240 ドット・32768 色・2プレーン

・[PF11] キーを押す ⇩

640×480 ドット・32768 色・1プレーン

・[PF11] キーを押す

● [PF12] キーによりMS-DOS互換画面に戻す

[PF12] キーを押すと、どのモードからでもMS-DOS互換画面に切り換えることができます。

図3.3 画面の切り換え

【制限事項】

①テキスト情報は、ページ0画面に表示されます。したがって、ページ0を使用したプログラムでは画面が乱れます。

②ページ1画面が前面に表示されている場合は、テキスト情報は表示されません。

③プログラムにおいてキーボードの初期化を行っている場合には、画面の切り換え（[PF11] ／ [PF12]）が無効になります。

386｜DEBUGを終了するには、“q”を入力します。

■端末使用によるデバッグ

FMTOWNS以外に端末（パソコンなど）をお持ちの方は、その端末を使用してデバッグ作業を行うことができます。端末を使用することにより、FMTOWNS単体でのデバッグ時の制限事項が解決されます。

●使用方法

① FMTOWNSとお持ちの端末（パソコンなど）のＲＳ－２３２Ｃポートを別売の“コンピュータ直結ケーブル”で接続します。

② 端末側で“ＴＥＲＭ”を実行します。設定は次のとおりです。

ボーレート　：９６００ｂｐｉ（端末による）
データ長　　：８ｂｉｔ
ストップ長　：２ｂｉｔ
パリティ形式：Ｎｏｎｅ
ＪＩＳコード：８ｂｉｔ
半／全二重　：Ｆｕｌｌ（全）
オートＬＦ　：Ｏｆｆ

③ 開発環境上から、“ＣＴＴＹ　ＡＵＸ”を入力します。

これで、テキスト情報は端末側に出力されるようになります。なお、デバッグ中の画面切り換えプログラムの使用方法は、FMTOWNS単体のデバッグ時と同様です。

●端末使用の終了

端末の使用を終了するには、端末側から、“ＣＴＴＹ　ＣＯＮ”を入力します。

これで、テキスト情報はFMTOWNS側に出力されるようになります。

3.5 動作確認

作成されたファイルは、それぞれの環境（システム）により実行方法が異なります。

【開発環境】

“.EXP”ファイルは、RUN386　<ファイル名>を使用します。

“.EXE”ファイルは、<ファイル名>をダイレクト入力します。

【ＭＳ－ＤＯＳ】

“.EXE”ファイルは、<ファイル名>をダイレクト入力します。

【ＴｏｗｎｓＯＳ】

①TownsMENU上で、ファイルをダブルクリックします。

②アイテム登録し、実行します。

アイテム登録されたプログラムは、TownsMENUで表示され、このアイテムをマウスで選択することにより、プログラムを実行することができます。アイテム登録方法の詳細は、FMTOWNSシステムソフトウェアに添付しているマニュアルを参照してください。

3.6 プログラミング手順 —— 例

「386｜ASM TOOL KIT」を使う

プログラムのアセンブルとリンク

プログラムのアセンブル、リンクに関する詳細は、「386｜ASM TOOL KITユーザーズマニュアル」に記述されています。ここではプログラム作成までの手順の例を簡単にご紹介します。

この例では、HELLO.ASM というプログラムを作成します。このプログラムはメッセージ "Hello world!" を表示するアセンブラ言語プログラムです。

●ソースプログラム例

```
cgroup  group   code
dgroup  group   data
        assume  cs:cgroup,ds:dgroup,ss:stack
code    segment dword public ER use32 'code'
print   proc    near
        lea     edx,hellomsg    ; call DOS to
        mov     ah,9            ; print string
        int     21h             ;
        mov     ax,4C00h        ; call DOS to
        int     21h             ; exit
print   endp
code    ends
data    segment byte public RW use32 'stack'
hellomsg db     'Hello world!',0Dh,0Ah,'$'
data    ends
stack   segment dword stack RW use32 'stack'
        db      1000h dup (?)
stack   ends
        end     print
```

●アセンブラの指定例

```
386ASM hello -twocase
```

●リンカの指定例

```
386link hello -twocase
```

リンク時に、-stackを指定しない場合はデフォルトで2000Kbyte となります。
実行ファイル（HELLO.EXP）を確認する場合は、開発環境上で実行させるか、TownsMENU上でアイテム登録して実行してください。

「High C™ Compiler」を使う

プログラムのコンパイルとリンク例

「プログラマーズガイド」でプログラムのコンパイル、リンクに関する詳細を記述していますが、簡単な操作例を次に紹介します。この場合、リンカは別売の「386|ASM TOOL KIT」に含まれている“386|LINK”を使用します。

例として、HELLO.Cプログラムを作成します。このプログラムはメッセージ“Hello world!”を表示するC言語プログラムです。

●ソースプログラム例

```
extern int printf();

void main()  {
    printf("Hello world!");
}
```

●コンパイラの指定例

```
hc386 hello
```

●リンカの指定例（386LINK使用）

```
386link hello -lib hce -twocase
```

実行ファイル（HELLO.EXP）を確認する場合は、開発環境上で実行させるか、TownsMENU上でアイテム登録して実行してください。

3.7 TownsMENU利用上の注意

アプリケーション終了時の注意

TownsMENUからアプリケーションを起動した場合は、アプリケーションが終了してTownsMENUに戻るときに、起動したドライブにTownsMENUの実行ファイル（TMENU.EXP）がないと、TownsMENUに復帰できません。

次のサンプルプログラムを参考に、必ずTownsMENUの実行ファイル（TMENU.EXP)があることを確認してください。なお、「386 | ASM TOOL KIT V1.1」に以下のオブジェクトファイルとサンプルとして動作確認プログラム(CHECK.C)、リンクファイル(TMENU.LNK)を提供しています。

リンク時は、TMENUCHK.OBJを先に指定してください。

【注意事項】
- ・リアルプログラムからの使用はできません。
- ・TownsMENU用アプリケーションでない場合は、リンクする必要はありません。

《TMENUCHK.OBJ》

```
                TITLE   TMENUCHK - アプリケーション終了処理

                                         /COPYRIGHT FUJITSU LIMITED 1989
                public  TMENU_chk
                public  _realend
                extrn   errno:Dword

RMCODE          SEGMENT WORD PUBLIC 'RMCODE' USE16
                ASSUME  CS:RMCODE
;**********************************************************************
;*      << ROUTINE-NAME : Set_interrupt_vector >>                     *
;*      (FUNCTION) :  割り込みベクタの設定( リアル)                     *
;*      (INPUT)                                                       *
;*      (OUTPUT)                                                      *
;*      (UPDATE HISTORY)                                              *
;**********************************************************************
            align   4
                public  SetVector
                extrn   _abort:NEAR
                extrn   _doserror:NEAR
SetVector       proc    far
                push    cs
                pop     ds
                mov     dx.offset _abort    ; ^C 割り込みの設定
                mov     ax.2523H
                int     21H
```

（次ページへ続く）

（前ページの続き）

```
                mov     dx,offset _doserror; ERROR 割り込みの設定
                mov     ax,2524H
                int     21H
                ret
SetVector       endp

;*****************************************************************
;*      << ROUTINE-NAME : MENU_abort >>                          *
;*      (FUNCTION) :  ^C キー割り込みの処理（リアル）            *
;*      (INPUT)                                                  *
;*      (OUTPUT)                                                 *
;*      (UPDATE HISTORY)                                         *
;*****************************************************************
            align   4
                public  _abort
_abort          proc    far
                iret
_abort          endp
;*****************************************************************
;*      << ROUTINE-NAME : MENU_abort >>                          *
;*      (FUNCTION) :  致命的エラー割り込みの処理（リアル）       *
;*      (INPUT)                                                  *
;*      (OUTPUT)                                                 *
;*      (UPDATE HISTORY)                                         *
;*****************************************************************
            align   4
                public  _doserror
_doserror       PROC    FAR
                MOV     BP,SP
                MOV     AX,DI
                ADD     AX,19
                OR      WORD PTR 1CH[BP],1   ; SET CARRY FLAG
                MOV     WORD PTR 06H[BP],AX ; SET RESULT CODE
                ADD     SP,6
                POP     AX
                POP     BX
                POP     CX
                POP     DX
                POP     SI
                POP     DI
                POP     BP
                POP     DS
                POP     ES
                IRET
                MOV     AL,1
_doserror       ENDP

                assume  CS:RMCODE
```

（次ページへ続く）

（前ページの続き）

```
Get_Drive       proc    far                 ; リアルプログラム開始!
                mov     ax.00000000H    ; Control.exe をコールする
                int     0D9H            ; ... ES:BX にアドレスが返ってくる
                mov     AL.Byte Ptr ES:[BX-1] ; ベースドライブ番号を取り込む
                add     AL."A"              ; ＡＳＣＩＩコード化する
                mov     Byte Ptr CS:File_Name.AL ; ストア
                ret                         ; 終了
Get_Drive       endp

File_Name       db      ?
                db      ":¥TMENU.EXP".0
_realend        label   byte

RMCODE          ends
;*******************************************************************
;*      << ROUTINE-NAME : TMENU_chk          >>                    *
;*      (FUNCTION) : 指定ドライブにＴＭＥＮＵ．ＥＸＰがあるか調べる*
;*      (INPUT)   int TMENU_chk(void);                             *
;*      (OUTPUT)  関数値          == -1 存在しない                 *
;*                                ==  0 存在する                   *
;*                extern errno   ==   ＥＲＲＯＲ情報               *
;*      (UPDATE HISTORY)                                           *
;*******************************************************************
CODE            segment 'CODE' use32
                assume  cs:CODE
TMENU_chk       proc    near
                push    offset Get_Drive        ; ドライブのベースアドレスを
                call    RealCall                ; 得るためのリアルをコールする
                pop     ecx                     ;
                mov     ax.00004300H          ; ファイルが存在するかをチェック
                mov     edx.offset File_Name  ; する
                int     21H                     ;
                jnc     TMENU_noerr            ;  エラーなしならばＯＫ
                mov     errno.eax             ; エラーならばエラーコードを返す
                mov     eax.-1                  ;
                jmp short TMENU_comm            ; 終了
TMENU_noerr:
                mov     errno.0                 ; エラーなしの処理
                mov     eax.0                   ;
TMENU_comm:
                ret                             ; 終了
TMENU_chk       endp
CODE            ends
                end
```

【注意】FMTOWNSで、アプリケーション起動用のバッチファイルをTownsMENUから起動する場合、環境変数用のコマンド（SET コマンド）の設定はできません。

リアルモードプログラムの呼び出し

次に、リアルモードプログラム呼び出しのためのサンプルプログラムを示します。

《REALCALL.OBJ》

```
             TITLE   REALCALL - ﾘｱﾙﾓｰﾄﾞﾌﾟﾛｸﾞﾗﾑの呼び出し
                                     COPYRIGHT FUJITSU LIMITED 1989

CODE    SEGMENT DWORD PUBLIC 'CODE' USE32
                ASSUME  CS:CODE
;************************************************************
;*      << ROUTINE-NAME : Call_real_program >>              *
;*      (FUNCTION) : リアルプログラムのコール                  *
;*      (INPUT)  RealCall(Address);                          *
;*      (OUTPUT)                                             *
;*      (UPDATE HISTORY)                                     *
;************************************************************
                align   4
                public  RealCall
                extrn   _realend:BYTE   ;リアルの最終アドレスをユーザが
                                        ; 設定して PUBLIC を宣言する.
RealCall        proc    near
                enter   0.0
                push    esi
                push    edi
                push    ebx
                push    es
                mov     bx.8[ebp]
                push    bx
                push    ds
                pop     es
                xor     ebx.ebx
                mov     ecx.offset_realend
                mov     ax.0250fh
                int     21h
                mov     ebx.ecx
                pop     bx
                mov     ecx.0
                mov     ax.0250eh
                int     21h
                pop     es
                pop     ebx
                pop     edi
                pop     esi
                leave
                ret
RealCall        endp
CODE            ends
                end
```

第4章 BIOSとソフトウェアライブラリ

4.1 概要

FMTOWNSでは、グラフィック機能、サウンド機能、CD-ROM機能などハードウェア依存の部分を「Towns BIOS」として提供しています。また、それらのC言語インターフェイスを「Cライブラリ」として提供しています。

4.2 Towns BIOSについて

FMTOWNS用のBIOSとしては、次の6種類のグループに分類して提供されています。

・グラフィックBIOS
・マウスBIOS
・スプライトBIOS
・フォントBIOS
・CD-ROM BIOS
・サウンドBIOS

グラフィックBIOS

FMTOWNSでは、グラフィックBIOSのことをEGB (Enhanced Graphic Bios) と呼びます。EGBの機能としては、次の4種類が提供されています。

セッティングオペレーション

EGB に関するハードウェアのパラメータを設定する機能（たとえば、どのような色で表示するかを指定する、パレット機能など）があります。

ブロックオペレーション

画面上の矩形域内のイメージデータを主記憶上に保存したり、その保存したデータを再表示する機能や、画面上のイメージデータを他の位置へ複写する機能があります。

グラフィックオペレーション

図形の大きさや位置を指定して描画する機能（たとえば、線分はその両端点の座標を指定して描画したり、円はその中心座標と半径を指定して描画する機能）があります。

フォントオペレーション

文字の表示に関する機能、文字列の描画の機能、文字間隔や文字描画方向を指定する機能などがあります。

マウスBIOS

マウスBIOSは、ポーリング動作の機能をもっています。また、割り込みハンドラを接続することにより、従来と同様な割り込み動作をすることができます。

インテリジェントマウスは、２ポートに対応し、接続されているポートで動作し始めます。auto_selectの機能や、マウスの移動距離や、その加速度を検出して移動距離を変化させる、加速度検出機能が内蔵されています。

マウスの動作は、EGBと連動させることによりすべての画面モードに対応し、またカラーのマウスカーソルを表示することもできます。

スプライトBIOS

スプライトBIOSは、スプライト画面の初期化、スプライト画面の表示、スプライトの定義、パレットブロックの設定、位置指定、アトリビュート設定、移動指定など、いろいろなスプライト制御を行うことができます。

フォントBIOS

フォントBIOSは、ANKフォントや漢字フォントの読み出し、SJISからJISへの変換、JISからSJISへの変換など、フォントに関する機能をもっています。

CD-ROM BIOS

CD-ROM BIOSは、CD-ROMに関する多くの機能をもっています。

ディスクドライブモードの設定、読み取り、ステータス情報読み取り、音楽演奏スタート、音楽演奏情報の読み取りなど多数の機能があります。

サウンドBIOS

サウンドBIOSは、主に音色の設定、発生、効果やタイマなどの制御を行います。一度に管理する音色数は、ＦＭ音源は 128音色、ＰＣＭ音源は32音色で、最大同時発生数はＦＭ音源が６音、ＰＣＭ８音源をサポートしています。

FM TOWNSには、ＦＭ音源ＬＳＩおよびＰＣＭ音源ＬＳＩが搭載されているため、これらをコントロールするためのドライバは、３つのレベルに分かれます。ユーザは、これら３つのどのレベルからでもアクセス可能です。

最も低レベルなルーチンは、音色変更や発音などの各種のコントロールを行う、サウンドBIOSです。次のレベルは、アセンブリやＣ言語のライブラリとしてMIDIエミュレータが提供されています。このMIDIエミュレータは、演奏ドライバが各音源をMIDI機器とみなしてコントロールできるようにするためのものです。これによりMIDIカードなどに対応して作られるアプリケーションで容易に内蔵音源に対応することができます。

最も高いレベルでは、 EUPHONY互換データ形式の演奏ドライバがあります。このデータ形式は、従来のＭＭＬなどより解像度が高く、トラック数も大幅に増えるため、よりきめ細かく、かつ重厚なサウンドをクリエイトできます。また、データの作成には EUPHONYを用いることができ、データの生産性が格段に向上します。

4.3 Ｃライブラリについて

FM TOWNSのＣライブラリは、FM TOWNSのＣ言語用のインターフェイスで、次の８種類のグループに分類して提供されています。

・グラフィックライブラリ
・マウスライブラリ
・スプライトライブラリ
・フォントライブラリ
・CD-ROMライブラリ
・サウンドBIOSライブラリ
・ EUPHONYドライバ
・キーボードインターフェイスライブラリ

各ライブラリは、前述したBIOSとほぼ同等の機能をもっていますので、参照してください。

FMTOWNS

アプリケーション開発キット
ユーザーズガイド
81SP-0071-1-0

発 行 日　1989年10月
発行責任　富士通株式会社

Printed in Japan

Ⓠ9005-6